APPLICATION FOR UNITED STATES PATENT

Inventor(s): John M. Haltmeyer
7535 Flamewood Drive
Clarksville, MD 21029
U.S. Citizen

Invention: **PRINTER MANAGEMENT PROTOCOL**

LAW OFFICES OF ROYAL W. CRAIG
210 N. Charles St.
Suite 1319
Baltimore, Maryland 21201
Telephone: (410) 528-8252

PRINTER MANAGEMENT PROTOCOL

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application derives priority from U.S. Provisional Patent Application No. 60/161,239 for "PRINTER MANAGEMENT PROTOCOL"; Filed: October 22, 1999.

BACKGROUND OF THE INVENTION

1. Field of the invention

The present invention relates to a printer management protocol for network printers and, more particularly, to a to a printer management protocol to assign and manage local and network printers in a networked computer environment.

2. Description of the Background

A primary goal of local area network (LAN) technology is to share all resources that are distributed across the network. Existing network software helps network administrators to accomplish this purpose. For example, Microsoft® Windows NT® Server 4.0 allows administrators to give anyone in their organization access to resources available on a Windows NT Server system, regardless of the client operating system they are using and without changing client software. Thus, no matter what operating system clients are equipped with, they can all connect to Windows NT Server using the protocols that they support natively. Consequently, most network resources can be managed from one location (centralized management). Resources like drive shares, SQL servers, mail servers, routers, etc. can be managed from remote locations. There is one resource that remains very difficult to manage. Specifically, printer management in

networks is still an elusive goal. This is primarily due to the complexities in managing the diverse operating systems of the client computers, and the numerous printer drivers required for each different type of printer and for each different operating system. Printers that are attached to the user's computer are not easily installed, configured or removed. In order to install and use a new printer on a user's computer, the device driver must be installed from the computer itself, the port must be configured and the device mode settings (things like page size) must be manually set.

As an example, FIG. 1 is a local area network diagram that helps to illustrate the obstacles faced in central printer management. Within the LAN a plurality of clients, e.g., 20-1-1..20-m-n are connected to respective network Servers 30-1...m via any of a number of different communication topologies inclusive of 10-base T, 100-base T, Sonet, Token Ring, etc. Clients 20-1..20-n might be configured to communicate with Server 30 via any of a number of different communication protocols such as IPX/SPX, TCP/IP, etc. In addition, Clients 20-1-1..20-m-n might be running any of various operating systems such as MacIntosh, Windows 95, 98 or Windows NT Workstation. Moreover, any number of different printers 40-1-1...40-m-n may be connected to the LAN, each requiring a different printer driver specially configured depending on the above constraints. If there are 100 client stations 20-1-20..20-5-20 each with printers attached, and there are 5 servers 30-1..5, then there would need to be 500 printer configurations defined and maintained. The management of client printer configurations has traditionally been a manual process and all printer drivers must be manually installed and network connections manually established. This is accomplished by the administrator physically working with each client station connected to the LAN, and the impracticality of the situation becomes apparent.

It would be greatly advantageous to provide a method for dynamically centrally configuring and managing a user's printer environment based on group membership, user name, terminal name or computer name.

SUMMARY OF THE INVENTION

In accordance with the above, it is an object of the present invention to provide a Printer Management Protocol (PMP) to dynamically centrally configure and manage a user's printer environment based on group membership, user name, terminal name or computer name.
In accordance with the above, it is an object of the present invention to provide a robust network printer management protocol that incorporates a single user interface for the assignment and management of printers and printer connections for users on a computer network.

It is another object to allow administrators to install and configure printers easily by maintaining a library of print drivers on a centralized server, or a synchronized farm of servers, for automatic real-time client printer driver installation and configuration regardless of operating system.

It is a further object to allow the installation and configuration of printers and printer connections from a remote / centralized location.

It is a further object to allow the simplified management of user's printers on a multi user computer system such as Microsoft Windows NT Terminal Server Edition, with or without Citrix MetaFrame installed.

In accordance with the above described objects, the present invention provides a system and method for assigning and managing the configuration of user's printers based on group

membership, user name, terminal name (client name) and computer name. The method can be implemented in any computer network and generally comprises the following steps:

1. Define a user's group membership, user name, terminal name and computer name. These definitions are known as "Owners", e.g., the combination of groups that a user belongs to as defined by group memberships, user name, terminal name (client name) and the computer name.
2. Query a database to determine which printers are to be assigned to the user.
3. Recursively execute a routine to configure local printers (connected to the user's computer or terminal), automatically install the required driver software and set the permissions of the printer, so only the targeted user's can access the printers.
4. Recursively execute a routine to connect to any assigned network printers (not directly connected to the use
5. Automatically set the default printer for the user.
6. Clean up all created printers and printer connections when the user logs off. This prevents the accumulation of unwanted/unneeded printers or printer connections.

BRIEF DESCRIPTION OF THE DRAWINGS

Other objects, features, and advantages of the present invention will become more apparent from the following detailed description of the preferred embodiment and certain modifications thereof when taken together with the accompanying drawings in which:

FIG. 1 is a perspective block diagram of an exemplary local area network (LAN) incorporating the printer management program (PMP) according to the present invention.

FIG. 2 is a top-level flow diagram of the PMP Client according to the present invention.

FIG. 3 is a detailed flow diagram of the Add Local Printers step 130 of Fig. 3 that displays the method for reading a database and adding local printers (printers directly connected to a user's computer or terminal).

FIG. 4 is a detailed flow diagram of the Add Network Printers step 140 of Fig. 3 that displays the method for reading a database and connecting to network printers (printers not directly connected to the user's computer or terminal).

FIG. 5 is a more detailed flow diagram of the Create Local Printer step 250 of Fig. 4 that shows the method for creating a local printer.

FIGs. 6.1-60 is a listing of an exemplary source code implementation of the printer management protocol of the present invention, inclusive of steps 110-470 (demarcated with step numbers indicated), written using Borlund® C++ Builder version 5.0.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention is a printer management protocol (PMP), or method, for automatically and centrally managing the printer environment for users on a computer network.

FIG. 1 is a perspective block diagram of an exemplary LAN capable of benefitting from the printer management protocol (PMP) according to the present invention. Within the LAN, each cluster of client stations, e.g., 20-1-1..20-1-n is connected to a network Server 30-1 via any of a number of conventional communication topologies inclusive of 10-base T, 100-base T, token ring,

etc. The Servers 30-1..m may be interconnected via an existing backbone in a distributed network. As previously described, a first cluster of Clients 20-1..20-n might be configured to communicate with their assigned Server 30-1 via any of a number of different communication protocols such as IPX/SPX, TCP/IP, etc. Clients 20-1..20-n might be running any of a number of different operating systems. In addition, any number of different printers 40-1-1...40-m-n may be connected to the LAN, each requiring a different printer driver specially configured depending on the above constraints. One or more printers may be locally attached to each client station (here printer 40-1-1 is local to client 20-1-1), and other printers may be connected elsewhere within the cluster or in other clusters and are remote. Thus, if the network administrator is using Client station 20-1-1, there is one local printer 40-1-1 attached as well as numerous other remote printers both in and out of the immediate cluster. Previously, there was no satisfactory solution for centralized printer management. All terminal printer connections needed to be manually configured on each Server 30-1...n in each cluster, and this led to hundreds and even thousands of printer configurations that had to be defined and maintained. The printer management protocol (PMP) according to the present invention automatically and centrally manages the configuration of the network printers based on group (cluster) membership, user name and client name (computer name).

The PMP protocol according to the present invention includes server software for centralized administration (herein called PMP Centralized Administration), and implementing client software called PMP Client, both of which are herein described. Both software packages can be developed for Windows operating systems. The Microsoft® Win32® application programming interface (API) allows applications to exploit the power of the 32-bit Windows

family of operating systems. Using the Win32 API, it is possible to develop applications that run successfully on all 32-bit versions of Windows. The Microsoft® Windows® graphics device interface (GDI) is the portion of the Win32® API that enables applications to use graphics and formatted text on the printer. Win32-based applications do not access the graphics hardware directly. Instead, GDI interacts with device drivers on behalf of applications. The GDI can be used in all Windows-based applications, and the Win32 API was designed for use by C/C++ programmers. Thus, both PMP Centralized Administration and PMP Client can be rendered in C/C++ programming language, and calls to the Windows API can be made directly from the respective programs. Microsoft® Windows® and Microsoft Windows NT®®/Windows 2000 provide a complete set of functions that allow applications to print on a variety of devices: laser printers, vector plotters, raster printers, and fax machines. The hundreds of Windows API functions and related structures are well-documented.

1. PMP Centralized Administration Server Software

The ability to centrally manage local and network printer assignment to computer users as well as provide the ability to dynamically install the required driver files requires the establishment of certain information constructs to facilitate the information flow needed by the PMP Client program to operate. First, a PMP database must be constructed to store information on all Owners, Local Printers, Network Printers, Assigned Local Printers, and Assigned Network Printers, and the PMP administration program must be capable of saving information to it. Second, a storage location must be established for the printer driver configuration files, and the

PMP Centralized Administration Server Software must be capable of saving printer device settings to the printer driver configuration files.

A. PMP Database Design

The PMP data structure should consist of five tables; Owners, LocalPrinters, NetworkPrinters, AssignedLocalPrinters, AssignedNetworkPrinters. The following is an example of a suitable layout of the five tables with descriptions of the purpose for the defined fields:

Owners

Field Name	Type	Size	Key	Description
ID	+		*	The unique identifier for the record.
Name	A	255		The name of the Owner. An Owner can be a local group, global group, user, client name or computer.
Ordinal	I			The number which specifies the order that the object should be placed. All queries are ordered by Ordinal.
TypeOfOwner	A	1		The type of Owner. L = local group, G = global group, U = user, T = terminal.

LocalPrinters

Field Name	Type	Size	Key	Description
ID	+		*	The unique identifier for the record.
Name	A	255		The name of the local printer. This name will be prefixed with <CLIENTNAME># or <USERNAME># if not running on Microsoft Windows NT Terminal Server Edition.
FileName	A	255		The name of the printer configuration file which stores the printer information.

Monitor	A	255		The printer port monitor name (Client Printer Port, Local Port, etc.).
Port	A	255		The name of the port (like "CLIENT\LPT!:", or 10.10.10.10::10.10.10.10)
SourceServer	A	255		The server that has the driver installed. PMP Client will automatically copy the driver files from this server if they do not exist on the client.
Disabled	L			True if this printer is disabled and should not be created.

NetworkPrinters

Field Name	Type	Size	Key	Description
ID	+		*	The unique identifier for the record.
Name	A	255		The UNC path for the printer (\\server\share).
Disabled	L			True if this printer is disabled and should not be connected to.

AssignedLocalPrinters

Field Name	Type	Size	Key	Description
ID	+		*	The unique identifier for the record.
OwnerID	I			The ID of the Owner table record to which the Printer is assigned.
LocalPrinterID	I			The ID of the LocalPrinters table record to which the Owner is assigned.
Map	A	255		Reserved.
IsDefault	L			True if this printer is to be the default for the user.
Ordinal	I			The number which specifies the order that the object should be placed. All queries are ordered by Ordinal.

AssignedNetworkPrinters

Field Name	Type	Size	Key	Description
ID	+		*	The unique identifier for the record.
OwnerID	I			The ID of the Owner table record to which the printer is assigned.
NetworkPrinterID	I			The ID of the NetworkPrinters table record to which the Owner is assigned.
Map	A	255		Reserved.
IsDefault	L			True if this printer is to be the default for the user.
Ordinal	I			The number which specifies the order that the object should be placed. All queries are ordered by Ordinal.

Again, the PMP Centralized Administration Server Software must be capable of storing the required information as specified in the above-described PMP Database Design tables.

B. Storage Location

In addition to the PMP Database, a storage location must be established for the printer driver configuration files. The storage location can be either centralized (stored on a UNC path) or distributed and synchronized (stored on each computer's local drive). The PMP administration program must be capable of saving printer device settings to a particular printer configuration file. The present invention accomplishes this using a developed TprinterControl::SaveLocalPrinter( ) function.

2. PMP Client

The compiled program that implements the present invention is the PMP Client. Under Windows NT, the PMP Client runs for each user in user mode. It could also be written to run as a service in system mode.

FIG. 2 is a top-level flow diagram of the PMP Client Software method according to the present invention.

At Step 100, the user logs on and PMP Client Software is launched from the UserInit string. UserInit is a system program that starts all programs listed in the UserInit string at logon and initializes variables. The implementing source code for step 100 can be found at FIG. 6.2.

Step 110 introduces a delay during which the program sleeps for a predetermined number of seconds (StartupDelay). The implementing source code for step 110 can be found at FIG. 6.9. In many cases, users will manually connect to network printers during their session. This can cause problems if the user is not familiar with the naming schemes involved in connecting to network printers. The PMP Centralized Administration Server Software allows the administrator to set a flag in the PMP Client Software to force the clearing of all network printers at logon. If this flag is set, Step 120 clears the network printer connections stored in the user profile. The implementing source code for step 120 can be found beginning at FIG. 6.10.

Program flow proceeds to Step 130 where local printers are added, and FIG. 3 illustrates a detailed Add Local Printers flow chart representative of this step. The implementing source code for step 130 can be found beginning at FIG. 6.3.

Beginning at Step 200, the program builds the select statement used to query the database for the assigned local printers. The implementing source code for step 200 can be found at FIG. 6.4. The select statement is the primary query command for Structured Query Language (also known as SQL). "Owners" are a combination of groups that a user belongs to, the user name, the terminal name (client name) and the computer name. Owners are assigned printers or printer connections in the PMP Centralized Administration program and these assignments are stored in the AssignedLocalPrinters table of the PMP Database described above.

Step 210 adds the Owners that the user belongs to, to the SQL select statement as shown. The implementing source code for step 210 can be found at FIG. 6.4.

Step 220 runs the query defined by the SQL select statement. The implementing source code for step 220 can be found at FIG. 6.4.

If there are no records returned, the Add Local Printers Step 130 is complete. Otherwise, for each record returned, the following steps are repeated.

Step 230 moves the database pointer to the next printer to be added. The implementing source code for step 230 can be found at FIG. 6.4.

Step 240 creates a new TPrinterControl object that encapsulates the printer creation. The implementing source code for step 240 can be found at FIG. 6.4.

Step 250 takes us to the Create Local Printer step, and FIG. 4 illustrates a detailed Create Local Printer flow chart representative of this step. The implementing source code for step 250 can be found beginning at FIG. 6.4.

With reference to FIG. 4, the creation of local printers is based on storing the printer information in a group of files called Printer Configurations.

Step 400 reads the printer configuration stored for the selected printer and populates the SelectedPrinterInfo structure (see TPrinterControl::SelectedPrinterInfo for details). The implementing source code for step 400 can be found at FIG. 6.38.

Step 410 sets the new printer name. The implementing source code for step 410 can be found at FIG. 6.38. If the printer already exists, Create Local Printer exits. A printer port monitor is the driver that controls the particular port that will be used to connect to the printer. Example: If the port is an LPT port, then the port monitor is a "Local Port". Step 420 validates the port monitor. The implementing source code for step 420 can be found at FIG. 6.38.

Step 430 validates the port, creating it if it does not exist. The implementing source code for step 430 can be found at FIG. 6.39. Step 440 validates the printer driver. The implementing source code for step 440 can be found at FIG. 6.39. If the print driver does not exist on the client computer, then PMP will install the driver automatically. Step 450 adds the new local printer. The implementing source code for step 450 can be found beginning at FIG. 6.39. Preferably, the new printer name is prefixed with either <CLIENTNAME># or <USERNAME># to maintain compatability with Citrix MetaFrame for Microsoft Windows NT Terminal Server Edition. The CLIENTNAME variable is defined as the name of the terminal connected to the Windows NT Terminal Server Edition and can be found by typing "SET" at the command prompt of the user.

Step 460 restores the printer settings saved in the assigned printer configuration file. The implementing source code for step 460 can be found beginning at FIG. 6.39.

Step 470 sets the permissions on the printer, so only the SYSTEM and the user have access to the printer, thus restricting the printer from unauthorized users. The implementing source code for step 470 can be found beginning at FIG. 6.40.

At this point, the creation of the new local printer is complete.

Returning back to the Add Local Printers routine of FIG. 3, if the printer has been flagged by the administrator (In the database) as the default printer, and a default printer has not yet been set, the method proceeds to Step 260.

Step 260 sets the printer as the default. The implementing source code for step 260 can be found at FIG. 6.5. Step 270 displays any error messages generated by Step 250. The implementing source code for step 270 can be found at FIG. 6.5.

Referring back to FIG. 2, once all local printer records have been processed, the Add Local Printers Step 130 is complete. After the local printers are added, program flow proceeds to Step 140, the Add Network Printers step. FIG. 5 illustrates a detailed Add Network Printers flow chart representative of this step.

Referring to FIG. 5, Step 300 builds the SQL select statement used to query the database for the assigned network printers. The implementing source code for step 300 can be found beginning at FIG. 6.6.

Step 310 adds the Owners that the user belongs to, to the SQL select statement. The implementing source code for step 310 can be found at FIG. 6.6.

Step 320 runs the query. The implementing source code for step 320 can be found at FIG. 6.6.

For each record returned, Step 330 moves the database pointer to the next printer to be added. The implementing source code for step 330 can be found at FIG. 6.6.

Step 340 adds the network printer connection using the AddPrinterConnection API call. The implementing source code for step 340 can be found at FIG. 6.6. If the printer has been

flagged by the administrator as the default printer, and a default printer has not yet been set, program flow proceeds to Step 350.

Step 350 sets the printer as the default, and the implementing source code for step 350 can be found beginning at FIG. 6.6.

Step 360 displays any error messages generated. Add Network Printers is complete. The implementing source code for step 360 can be found beginning at FIG. 6.7.

Referring back to FIG. 2, once all printers have been created and connected, program flow proceeds to step 150 and the PMP Client Software will wait for the WM_ENDSESSION message to be broadcast. The implementing source code for step 150 can be found beginning at FIG. 6.12.

Once the WM_ENDSESSION message has been received, Step 160 removes the printers and printer connections made by PMP Client. The implementing source code for step 160 can be found beginning at FIG. 6.9.

Step 170 is the logoff of the user and closing of PMP Client Software. The implementing source code for step 170 can be found beginning at FIG. 6.2.

The above-described client and server software components and the methods reflected therein make it possible to map printers back to the client stations 20-1-1...n. By assigning printers 40-1-1...n to the ClientName (terminal name), the PMP is able to automatically create the printer queues as the users logon from those client stations. Printers are also created additively based on Windows NT group membership. So, if a particular group such as the accounting department has a printer that has specific settings, the PMP can configure all authorized user connections to this printer simply by assigning the printer to the accounting group with all the

correct properties, and then assigning the appropriate users to the accounting group. Printers can also be denied to groups of users, users and clientnames (computer names).

The method is ideal for shared printing in mixed-LAN environments, and stands as a solution for all major network operating systems for Ethernet, 100Base-T, 100VG AnyLAN, Token Ring, and LocalTalk networks. It facilitates easy installation and configuration of virtually any printer for direct LAN connectivity. The entire process is automated, and all printer management protocol and printer configuration problems are solved. It is simple for LAN administrators to remotely configure and manage all printers. There is automatic driver installation and printer configuration, and remote configuration is shown graphically on the administrator PC screen. The protocol greatly simplifies printer management over PC print servers, file server connections and PC parallel-port connected printers, inclusive of high-speed IEEE 1284 compliant parallel ports on external print servers.

Having now fully set forth the preferred embodiments and certain modifications of the concept underlying the present invention, various other embodiments as well as certain variations and modifications of the embodiments herein shown and described will obviously occur to those skilled in the art upon becoming familiar with said underlying concept. It is to be understood, therefore, that the invention may be practiced otherwise than as specifically set forth in the appended claims.